
Tutorial: Sinusoidal Wall Temperature Variation

Introduction

This tutorial examines fluid flow through a two-dimensional channel, where one wall of the channel has a user-defined temperature profile applied to it.

The purpose of this tutorial is to demonstrate the ability of **FLUENT** to use user-defined functions (UDFs) to specify a position-dependent variable on the wall boundary condition.

Prerequisites

This tutorial assumes that you are familiar with **FLUENT** interface and that you have a good understanding of the basic setup and solution procedures. If not, the **FLUENT** Tutorial Guide will provide you with the necessary experience. If you have not used UDF's, it would be helpful to refer to the **FLUENT** UDF Manual.

Problem Description

The schematic of the problem considered in this tutorial is shown in Figure 1.

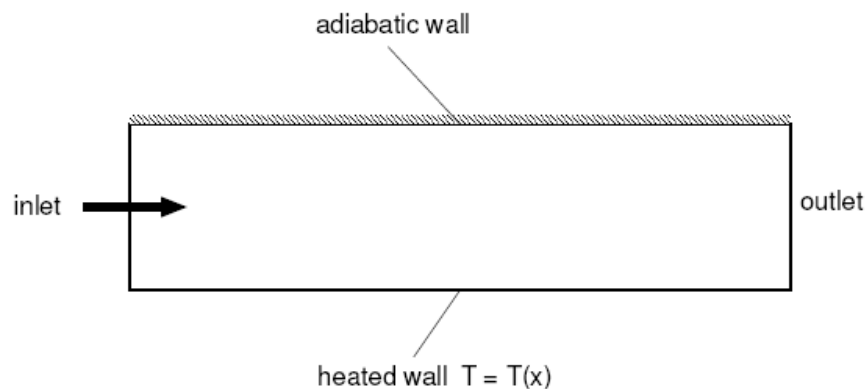


Figure 1: Problem Specification

Air at 300 K enters a 2-D channel which has an insulated upper wall and a heated lower wall.

The temperature on the lower wall varies sinusoidally with x-position according to:

$$T(x) = 300 + 100 \sin\left(\frac{\pi x}{0.005}\right)$$

The calculation will be performed assuming steady state, incompressible, laminar flow in the channel.

Preparation

1. Copy the files `channel.msh` and `wallprof.c` to your working folder.
2. Start the 2D (2d) version of FLUENT.

Setup and Solution

Step 1: Grid

1. Read the grid file `channel.msh`.
2. Check the grid.
3. Display the grid.

Grid → Check

Display → Grid...

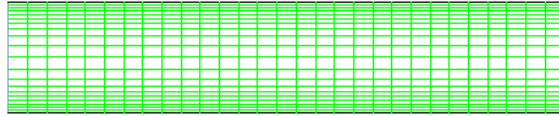


Figure 2: Graphics Display of the Grid

Step 2: Models

1. Retain the default solver settings.
2. Enable energy equation.

Define → Models → Energy...

Step 3: Materials

You will use the default fluid properties of air for this problem. Therefore, you need not make any changes to the material properties.

Step 4: Interpret the UDF

1. View the contents of the UDF.

The user defined function wallprof.c is used to specify a sinusoidal temperature variation on the wall. In a separate editor, you can view the contents of the UDF to understand its structure and function. The contents of the UDF file are as follows:

```
/* **** */
/* User-Defined Functions for sinusoidal temperature profile */
/* **** */
/* **** */

#include "udf.h"
#define PI 3.141592654

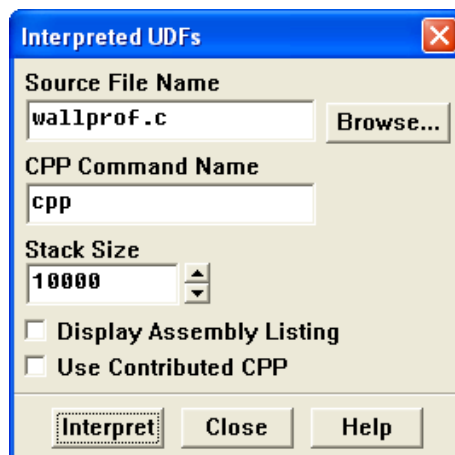
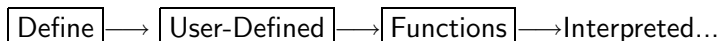
DEFINE_PROFILE(temperature_profile, thread, position)
{
    real r[3]; /* this will hold the position vector */
    real x;
    face_t f;

    begin_f_loop(f, thread)

        F_CENTROID(r,f,thread);
        x = r[0];
        F_PROFILE(f, thread, position) = 300.+100.*sin(PI*x/0.005);

    end_f_loop(f, thread)
}
```

2. Interpret the UDF.



- (a) Click the Browse... button.
- (b) Select the source file `wallprof.c` in the Select File panel.
- (c) Specify the C preprocessor to be used in the CPP Command Name field.
If you want to use the C preprocessor that Fluent Inc. has supplied, instead of using your own, you can select the Use Contributed CPP option.
- (d) Retain the default value of 10000 for Stack Size, unless the number of local variables in your function will cause the stack to overflow.
The Stack Size should be set to a number that is greater than the number of local variables used.
- (e) Click Interpret and close the Interpreted UDFs panel.

For more information on interpreted UDFs, refer to the UDF Manual.

Step 5: Boundary Condition

Define → Boundary Conditions...

1. Set the boundary conditions for the wall-1 zone.
 - (a) Click the Thermal tab and select Temperature from the Thermal Conditions list.
 - (b) Select `udf temperature_profile` from the Temperature drop-down list.
 - (c) Retain the default values for the other parameters.
 - (d) Click OK to close the Wall panel.
2. Set the boundary conditions for the velocity-inlet-1 zone.
 - (a) Select Components from the Velocity Specification Method drop-down list.
 - (b) Enter 1 for X-Velocity.
 - (c) Click OK to close the Velocity Inlet panel.
3. Retain the default settings for the other zones.

Step 6: Solution

1. Initialize the flow field using the boundary conditions set at velocity-inlet-1.

Solve → **Initialize** → Initialize...

2. Enable plotting of residuals and change the Absolute Criterion for continuity to 0.0001.

Solve → **Monitors** → Residual...

3. Start by requesting 100 iterations (Figure 3).

The solution converges in approximately 60 iterations.

Solve → Iterate...

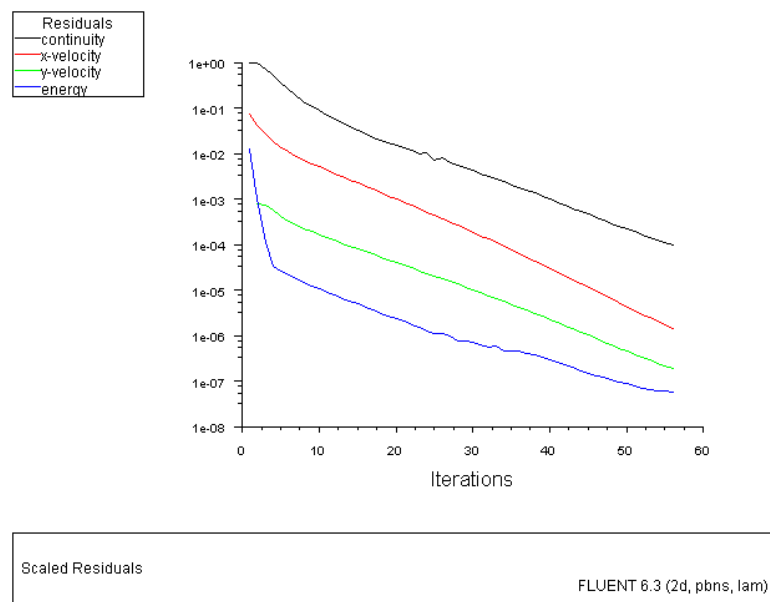


Figure 3: Scaled Residuals

Step 7: Postprocessing

1. Display filled contours of static temperature (Figure 4).

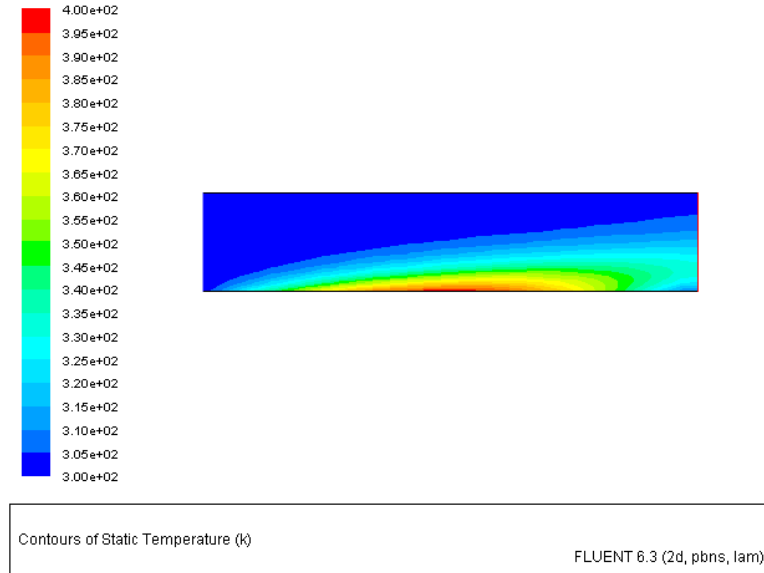


Figure 4: Contours of Static Temperature

Results

The contour plot in Figure 4 shows that the temperature on the wall and in the fluid reaches a peak at the center of the channel due to the peak in the prescribed wall temperature.

Summary

The tutorial has demonstrated the use of UDFs for specifying profiles on boundaries. This approach can be applied to the other boundary condition types such as pressure and velocity inlets, and pressure outlets.

When you are comfortable with this problem, try modifying the UDF to specify heat flux rather than temperature. This will require changing the line:

```
temp = 300.+100.*sin(pi*x/0.005);
```

with the specified heat flux expression in your udf code, and changing the boundary condition type as well as hooking the new profile to the wall boundary before running the new case.